LlamaTurk: Adapting Open-Source Generative Large Language Models for Low-Resource Language

Anonymous ACL submission

Abstract

Despite advancements in English-dominant generative large language models, further development is needed for low-resource languages to enhance global accessibility. The primary methods for representing these languages are monolingual and multilingual pretraining. Monolingual pretraining is expensive due to hardware requirements, and multilingual models often have uneven performance across languages. This study explores an alternative solution by adapting large language models, primarily trained on English, to low-resource languages. We assess various strategies, including continual training, instruction fine-tuning, task-specific fine-tuning, and vocabulary extension. The results show that continual training improves language comprehension, as reflected in perplexity scores, and task-specific tuning generally enhances performance of downstream tasks. However, extending the vocabulary shows no substantial benefits. Additionally, while larger models improve task performance with few-shot tuning, multilingual models perform worse than their monolingual counterparts when adapted.

1 Introduction

The performance of proprietary generative large language models (LLMs) is better than open-source ones in most cases as this article is written (Xu et al., 2022; Sun et al., 2024), though there are efforts to develop open-source generative LLMs in terms of high performance and human ethics alignment (Touvron et al., 2023a; Jiang et al., 2023; Almazrouei et al., 2023).

The progress is more significant in the English language compared to other languages as the aforementioned open-source models are mostly trained by English corpora (Wang et al., 2023; Zhang et al., 2023a). To make natural language processing technology more inclusive and accessible globally, research and development should be dedicated to the techniques that improve the performance of large language models in low-resource languages.

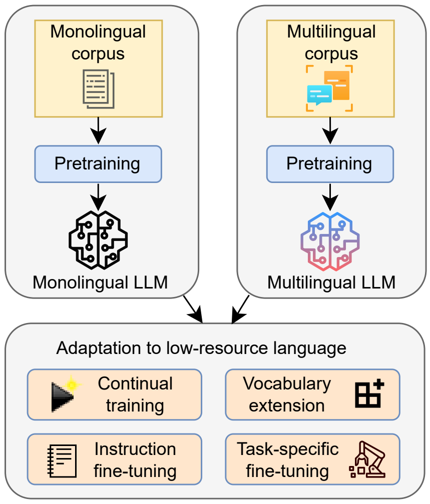


Figure 1: Adapting generative large language models for low-resource languages.

Monolingual (Yang et al., 2023b; Nagoudi et al., 2023; Uludoğan et al., 2024; Corrêa et al., 2024; Kesgin et al., 2024) and multilingual pretraining (Shliazhko et al., 2023; Scao et al., 2022; Lin et al., 2024b) of generative LLMs are two main solutions for representing low-resource languages. However, monolingual pretraining is too costly due to hardware requirements for generative LLMs (Zhao et al., 2023a). On the other hand, multilingual LLMs have uneven performance across different languages mostly due to imbalanced training corpus (Zhang et al., 2023a; Qin et al., 2024). Our proposed solution is to adapt open-source generative LLMs for low-resource languages, illustrated in Figure 1.

In this regard, this study examines how to adapt open-source LLMs for low-resource languages in a systematic way. We focus on the benefits of using different methodologies, both individually and together, including continual training, supervised fine-tuning, and vocabulary extension, to adapt generative LLMs for low-resource languages.

For the sake of efficiency, we use Llama (Touvron et al., 2023a) in the experiments. We select the Turkish language as a low-resource language. We therefore refer to the model family used in this study as `LlamaTurk`. The model size and language selection are affordable when the number of experiments is considered in this study[1]. Also, Llama is trained mostly with English data, which can provide better investigation for adapting non-English languages. The Turkish language can be categorized under low-resource languages when training corpus of open-source generative LLMs are considered (Touvron et al., 2023a), yet the recipes given in this study can also be used for other low-resource languages since the methods are independent of language itself.

We further examine adaptation in terms of two more aspects: Model size and multilinguality. Model size is important for scalability and performance (Zhao et al., 2023a; Yang et al., 2023a). We provide an analysis of the adaptation of Llama-7b and 13b in this respect. Moreover, multilingual LLMs, such as BLOOM (Scao et al., 2022), Yi (AI et al., 2024), Aya (Üstün et al., 2024), and MaLA (Lin et al., 2024a), can provide an opportunity to adapt low-resource languages easier than English-dominant ones due to multilingual corpus and vocabulary. Since BLOOM and Yi do not involve Turkish in training and Aya is larger than MaLA in terms of model parameters, we use MaLA for an analysis of multilingual LLMs.

The main contributions of this study can be summarized as follows. We (i) analyze the adaptation of generative LLMs for low-resource language systematically to understand advantages and disadvantages in terms of continual training, instruction fine-tuning, task-specific fine-tuning, and vocabulary extension, (ii) investigate model size and multilingual models for adaptation, and (iii) publish all resources including source codes, datasets, and generative models reported in the experiments[2].

2 Related Work

Generative LLMs are either proprietary or open-source. Although proprietary LLMs have currently outstanding performance (Sun et al., 2024), there are also efforts to develop competitive open-source models (Touvron et al., 2023a; Jiang et al., 2023).

The majority language of open-source generative LLMs is English. Their pretraining text corpus mostly includes text in the English language. For adapting LLMs pretrained with English data for low-resource languages, the following methods are examined. (i) The training phase is continued using non-English raw data to learn the language properties of the new language (Larcher et al., 2023; Cui et al., 2024; Zhao et al., 2024; Acikgoz et al., 2024). (ii) The knowledge of large language model is transferred by supervised fine-tuning on a non-English instruction or downstream-task dataset (Santilli and Rodolà, 2023; Holmström and Doostmohammadi, 2023; Kohli et al., 2023; Zhao et al., 2024; Garcia et al., 2024; Kuulmets et al., 2024). (iii) The vocabulary of large language model is extended to include non-English tokens (Cui et al., 2023; Zhao et al., 2024).

These methods are employed in different studies and languages, resulting in a lack of understanding advantages and disadvantages of each in a controlled experimental framework. Different from these studies, we provide a comprehensive experimental setup on the benefits of different methodologies for adapting generative LLMs for low-resource languages. Moreover, we focus on model size and multilingual models for adaptation.

3 Adaptation Methods

In this section, we explain the methods to adapt open-source generative LLMs for low-resource languages in detail.

3.1 Continual Training

Continual training is the process of extending the pretraining phase of LLMs by incorporating new data corpus (Gupta et al., 2023). The main objective is to minimize the loss on this new data while having relatively lower loss scores on previous data since continual training is open to catastrophic forgetting (French, 1999; Li and Lee, 2024). Continual training can therefore capture implicit language structures and text semantics.

Previous studies (Qin et al., 2022) show that continual training improves the performance of domain adaptation for BERT-like encoder-based LLMs (Devlin et al., 2019). It is also used for adapting decoder-based generative LLMs to low-resource (Cui et al., 2023; Zhao et al., 2024), code-mixed (Owen et al., 2024), non-Latin (Husain et al.,

[1] Two NVIDIA RTX 2080Tis and four A4000s are employed in the experiments.

[2] Anonymous

2024), and multilingual (Lin et al., 2024a) settings.

In this study, similar to previous studies, we employ Low-Rank Adaptation (LoRA) (Hu et al., 2021) for efficient training due to limited resources. We use a raw Wikipedia corpus[3] from November 2023 with a size of 534,988 Turkish articles.

We set the input sequence length as 512 tokens and the batch size as 128 instances. We use 32 gradient accumulation steps and 100 linear warmup steps. We train with a learning rate of 3e-4 for a single epoch. LoRA's R is set to 8, alpha to 16, and dropout to 0.05. Since continual training is costly and the study has a limited budget, we employ continual training for only Llama-7b[4] with 8-bit quantization. A single run of continual training takes approximately 206 hours with these settings using four NVIDIA RTX A4000s.

3.2 Instruction Fine-tuning

Instruction tuning is a supervised fine-tuning method that improves the ability of LLMs to follow instructions (Wei et al., 2021; Ouyang et al., 2022; Zhang et al., 2024). During training, the model is presented with many pairs of instructions and corresponding responses. The main objective is to teach the model to generate accurate responses based on the given instructions, rather than continuing from the previous text.

Different from previous instruction-tuning efforts, Stanford's Alpaca (Taori et al., 2023) is a leading model that shows major improvements by instruction fine-tuning an open-source generative LLM, namely (Touvron et al., 2023a). While Alpaca and similar models such as Vicuna (Chiang et al., 2023) have an instruction set constructed by prompting proprietary LLMs, other models such as Dolly (Conover et al., 2023) employ human labor for constructing a more reliable instruction set. The majority of these efforts are for the English language, yet there are instruction-tuned models to adapt English-supported LLMs for low-resource settings (Cui et al., 2023; Zhao et al., 2024; Azime et al., 2024).

In this study, we construct an instruction set by translating Alpaca's 52k instructions from English to Turkish by using Google Translate[5]. The quality of the translated set is inadequate for training since we observe many issues such as translation errors (e.g. missing letters and untranslated words), keyword translations (e.g. reserved keywords specific to programming languages should not be translated), and semantic mismatching (e.g. original instruction asks for a phrase with five words, but correct translation has less than five words). We therefore manually validate and correct the quality of the instruction set. We publish our instruction set[6]. We also provide a prompting example for instruction fine-tuning in Appendix A.1.

We employ instruction tuning for all LLMs examined in this study, namely Llama-7b[7], Llama-13b[8], and MaLA-10b[9]. We use 8-bit quantization with LoRA (resulting in training 12.4% of LLM parameters) and the same hyperparameters as in continual training, except that we use a smaller input sequence length (256 tokens) and train for two epochs. A single run of instruction tuning takes approximately 17.5 hours for Llama-7b with these settings using two NVIDIA RTX 2080Tis.

3.3 Task-Specific Fine-tuning

Task-specific tuning is a type of instruction tuning, where a fine-tuning set involves task-related instructions and ground-truth answers (Budzianowski and Vulić, 2019; Wang et al., 2024), rather than adapting a general-purpose instruction set. Task-specific tuning of generative LLMs is proven to be successful in different domains including text editing (Raheja et al., 2023), sentiment analysis (Inserte et al., 2024), and machine translation (Zheng et al., 2024). However, task-specific tuning have the potential of deteriorating the language capabilities of LLMs (Zhang et al., 2023b; Zhao et al., 2023b).

We follow instruction fine-tuning with a task-specific dataset for the downstream task of sentiment analysis. We choose sentiment analysis since it is a widely applicable task that represents a fundamental natural language processing capability (Liu, 2012). For this purpose, we create an instruction set for sentiment analysis. To create a balanced set, we downsample 2,500 instances for both negative and positive sentiment classes, a total of 5k instances from the TRSAv1 dataset (Aydoğan and Kocaman, 2023). We then use a prompt manually crafted for the task of sentiment analysis[10]. We provide the prompt in Appendix A.2.

[3]https://huggingface.co/datasets/wikipedia
[4]https://huggingface.co/huggyllama/llama-7b
[5]https://translate.google.com

[6]Anonymous
[7]https://huggingface.co/huggyllama/llama-7b
[8]https://huggingface.co/huggyllama/llama-13b
[9]https://huggingface.co/MaLA-LM/mala-500-10b-v1
[10]We run prompts from existing resources (Bach et al., 2022) but decided to use a manually crafted one by observing better performance in preliminary experiments.

	Data	Size	Tokens
Continual training	Wiki	535.0k	273.9m
Instruction tuning	Alpaca	52.0k	13.3m
Task-specific tuning	Sentiment	5.0k	1.3m
Vocabulary extension	BPE	28.6k	28.6k

Table 1: **Data statistics for adaptation methods**. The columns represent the type of data used (Data), the total number of instances (Size), and the total number of tokens (Tokens), respectively.

We employ task-specific tuning for all LLMs examined in this study. We use all models in 8-bit quantization. We also use LoRA (resulting in training 12.4% of LLM parameters) and the same hyperparameters as in instruction tuning. A single run of task-specific tuning takes approximately 2.5 hours for Llama-7b with these settings using two NVIDIA RTX 2080Tis.

3.4 Vocabulary Extension

Vocabulary embeddings are a major component of how LLMs understand and process natural language text by capturing semantic meanings and relationships among subwords called tokens. Vocabulary tokens are determined by tokenization algorithms such as WordPiece (Schuster and Nakajima, 2012) and Bype Pair Encoding (BPE) (Sennrich et al., 2016).

Llama has a vocabulary size of 32k tokens based on BPE tokenization (Touvron et al., 2023a). The majority of tokens in its vocabulary are English. The remaining small portion involves European languages with Latin and Cyrillic symbols.

In this study, we extend Llama's vocabulary by merging with low-resource language tokens. Specifically, we use the Turkish tokenizer with 28,600 tokens trained by BPE algorithm (We publish the tokenizer[6]).

Merging the original Llama tokenizer with low-resource vocabulary yields 59,773 tokens, meaning that 827 tokens are overlapping. This results in adding almost 228m new parameters to be trained into the model due to the extended vocabulary embeddings. We employ vocabulary extension with above-mentioned methods when Llama-7b is used with LoRA due to limited resources.

3.5 Combinations

A summary of data statistics used for the adaptation methods is given in Table 1. In addition to a single examination of these methods, we also report the results of using them in combination to leverage better performance. We particularly employ the following combinations using Llama-7b with LoRA. Hyperparameters are set the same as explained in the previous subsections.

Continual Training with Instruction Fine-tuning: We first obtain a model by continual training using raw Wiki data as explained in Section 3.1. We then apply instruction fine-tuning as explained in Section 3.2. The motivation is to boost the potential of instruction tuning when the backbone model is trained with low-resource raw text beforehand.

Continual Training with Task-Specific Fine-tuning: With a similar motivation to the previous approach, we first obtain a model by continual training using raw Wiki data. We then apply task-specific fine-tuning as explained in Section 3.3.

Continual Training with Instruction and Task-Specific Fine-tuning: The motivation is to boost the performance of task-specific tuning when the model is trained by both raw text and instruction-set in low-resource language beforehand. We first obtain a model by continual training using raw Wiki data. We then apply instruction tuning and task-specific fine-tuning respectively.

Instruction and Task-Specific Fine-tuning: This approach avoids continual training but examines using both instruction and then task-specific tuning respectively. The motivation is to boost the performance of task-specific tuning when the model is trained by only instruction-set in low-resource language beforehand.

Vocabulary Extension with Instruction Fine-tuning: We extend the vocabulary with low-resource language tokens as explained in Section 3.4. We then apply instruction tuning to understand the impact of vocabulary extension on instruction tuning.

Vocabulary Extension with Task-Specific Fine-tuning: With a similar motivation to the previous approach, we extend the vocabulary with low-resource language tokens and then apply task-specific tuning to understand the impact of vocabulary extension on task-specific tuning.

Vocabulary Extension with Continual Training: We extend the vocabulary with low-resource language tokens and then apply continual training to understand its impact on continual training.

4 Experiments

In this section, we evaluate the performance of different methods to adapt generative large language

	xquad question	xquad context	dbricks instruction	dbricks response
Size	1.2k	1.2k	15.0k	15.0k
Chars	74.7k	965.4k	1.1m	5.4m
Tokens	37.4k	458.3k	549.8k	2.4m

Table 2: **Dataset statistics for perplexity**. The xquad dataset has question and context subsets. The databricks (dbricks) dataset has instruction and response subsets.

models for low-resource language. We particularly conduct both intrinsic and extrinsic evaluations in order to understand the performance of both language comprehension and downstream tasks. We also run benchmark LLM evaluation by using appropriate datasets. This section further involves the results of using varying model sizes and applying multilingual models for the adaptation.

4.1 Intrinsic Evaluation

Intrinsic evaluation of generative LLMs involves a perplexity score that represents how well a language model can predict the next word in a sequence of text (Jurafsky and Martin, 2009):

$$\text{perplexity} = 2^{-\frac{1}{N}\sum_{i=1}^{N}\log_2 P(w_i|w_1,\ldots,w_{i-1})} \quad (1)$$

where N is the total number of words and $P(w_i|w_1, w_2, \ldots, w_{i-1})$ is the probability assigned by the model to the i-th word given the preceding text context.

A lower perplexity score indicates that language model is better able to predict the next word, and thus has a better understanding of the language.

We calculate the perplexity scores on different data collections than the ones used in Section 3. Specifically, we use the Turkish question and context subsets of `xquad` (Artetxe et al., 2019), and the instruction and response subsets of `databricks-dolly-15k` (Conover et al., 2023) using a Turkish translated version[11]. The detailed statistics of the data used for calculating perplexity scores are given in Table 2. The reason for reporting the perplexity scores for different subsets is that the characteristics of each subset can be helpful to understand the applied method's impact on the adaptation. For instance, `xquad-question` has instances of questions while `xquad-context` has longer paragraphs of task descriptions. Similarly, `databricks-instruction` has instruction-type questions, while `databricks-response` has answers or responses to those questions.

[11]https://huggingface.co/datasets/atasoglu/databricks-dolly-15k-tr

In Table 3, we provide the perplexity scores. The main observations can be summarized as follows.

Continual training reduces perplexity scores. In all cases, perplexity scores are improved by continual training (`LlamaTurk-7b-c`). The lowest perplexity scores are also obtained by continual training in the majority of cases (three of four data collections). A possible reason is that the model could gradually accumulate language knowledge as it is exposed to more raw text. This incremental learning process can allow the model to become more robust and adaptable.

Instruction tuning improves perplexity but not task-specific tuning. Perplexity scores are improved by instruction tuning (`LlamaTurk-7b-i`). The only exception is `xquad-context`, yer instruction tuning has still a very close perplexity score to the original `Llama-7b`. Our instruction-tuning set is based on Alpaca, which has general-purpose instructions and responses. On the other hand, task-specific tuning (`LlamaTurk-7b-t`) deteriorates perplexity scores in all cases. We argue that, by training on task-specific instructions, generative LLMs might become overly specialized and optimized for those specific instructions, rather than maintaining a more general understanding of language.

Combinations fail in most cases but depends on data types. The combinations that include task-specific tuning have poor perplexity scores. On the other hand, continual training and instruction tuning improve perplexity. We therefore expect to have a better performance by using them together (`LlamaTurk-7b-c-i`) but perplexity scores get worse than the case when they are applied alone. However, when perplexity is measured on an instruction set (`databricks-instruction`), continual training together with instruction tuning has the lowest perplexity score. This observation can support that generative LLMs adapt to different data types, and one should consider target data type before selecting adaptation method.

Vocabulary extension has poor perplexity. In all models where vocabulary extension is applied (`Llama-7b-v`), perplexity scores get higher than the original (`Llama-7b`). We argue that without sufficient training data and fine-tuning, the model can struggle to effectively incorporate the new vocabulary into its internal representations and learning

Model	Continual Training	Instruction Tuning	Task Tuning	Vocabulary Extension	Data			
					xquad question	xquad context	dbricks instruction	dbricks response
Llama-7b					6.6916	1.5487	9.5845	9.0259
LlamaTurk-7b-c	✓				**5.5088**	**1.5064**	8.4364	**7.0924**
LlamaTurk-7b-i		✓			6.3260	1.5674	8.3131	7.9351
LlamaTurk-7b-t			✓		9.2267	1.7850	13.7173	13.2289
LlamaTurk-7b-c-i	✓	✓			7.0676	1.5978	**8.2488**	9.4570
LlamaTurk-7b-i-t		✓	✓		9.0380	1.8194	13.0113	11.8501
LlamaTurk-7b-c-t	✓		✓		7.7305	1.7181	12.5591	10.7188
LlamaTurk-7b-c-i-t	✓	✓	✓		8.0855	1.6666	11.5441	10.6943
LlamaTurk-7b-v-i		✓		✓	18.6241	3.8897	22.1750	24.3312
LlamaTurk-7b-v-t			✓	✓	28.7707	5.8666	37.6394	43.7040
LlamaTurk-7b-v-c	✓			✓	17.3135	3.6807	23.9212	23.2612

Table 3: **Perplexity scores**. The models have different adaptation methods: Continual Training (c), Instruction Tuning (i), Task-specific Tuning (t), and Vocabulary Extension (v). The xquad dataset has question and context subsets. The databricks (dbricks) dataset has instruction and response subsets. The best (lowest) perplexity scores for each dataset are given in bold.

processes. Similarly, (Zhao et al., 2024) observes negative impact of vocabulary extension, and also suggests that vocabulary extension might not be a suitable choice for small-scale continual training such as in our continual training with 0.2 billion tokens of the training data. Another reason could be the number of additional tokens in vocabulary (28k tokens), merged with the original tokenizer (32k tokens). More experimentation is needed to understand if a different number of new tokens in vocabulary works better in adaptation.

4.2 Extrinsic Evaluation

Generative LLMs employ human evaluations as an evaluation method to align with human judgments (Ouyang et al., 2022). However, human-based evaluation is labor-intensive, making it costly and less feasible for low-resource languages. On the other hand, LLM evaluation benchmarks offer reliable evaluation for downstream NLP tasks such as GLUE (Wang et al., 2018) and SuperGLUE (Wang et al., 2019). Similarly, there are evaluation frameworks and tools such as LM Evaluation Harness (Gao et al., 2023) and MLflow[12]. However, they mostly support English benchmark datasets. Although multilingual datasets are published by some benchmarks, either they do not include the language used in this study, or the data size is small for task-specific tuning. We therefore craft an evaluation on sentiment analysis in this subsection[13].

For this purpose, we extract 100 instances (50 instances for both positive and negative classes) from the Turkish sentiment analysis dataset used in task-specific tuning (Aydoğan and Kocaman, 2023). We avoid selecting from 5k instances used in task-specific tuning explained in Section 3.3. Since inference is time costly, we use a small subset of this dataset for the evaluation. We also craft inference prompts for different scenarios including zero-shot to few-shot prompts. We check the generated text if it equals to positive or negative, and calculate the accuracy score accordingly. We measure accuracy since the inference dataset is fully balanced. We provide the inference prompts in Appendix A.3.

Model	0-shot	1-shot	2-shot	3-shot
Llama-7b	0.00	0.50	0.53	0.50
LlamaTurk-7b-c	0.00	0.47	0.54	0.51
LlamaTurk-7b-i	0.06	0.48	0.48	0.56
LlamaTurk-7b-t	0.90	0.84	0.61	0.78
LlamaTurk-7b-c-i	0.10	0.52	0.50	0.54
LlamaTurk-7b-i-t	0.83	0.90	0.93	0.89
LlamaTurk-7b-c-t	0.82	0.60	0.62	0.86
LlamaTurk-7b-c-i-t	0.62	0.52	0.56	0.51
LlamaTurk-7b-v-i	0.35	0.44	0.49	0.53
LlamaTurk-7b-v-t	0.44	0.50	0.53	0.53

Table 4: **Accuracy scores on sentiment analysis**. The darker cell color gets, the better task performance.

During inference, we load the models with 8-bit quantization due to limited hardware. Generation configuration involves the following hyperparameters. The temperature is set to 0.2. Beam search is applied with four beams, and top-p is set to 0.75. A single run of inference takes approximately from six hours (zero-shot) to eight hours (3-shot) for Llama-7b with these settings using two NVIDIA RTX 2080Tis.

In Table 4, we provide the perplexity scores for all methods. The main observations are as follows.

[12] https://github.com/mlflow/mlflow

[13] We also provide a benchmark evaluation for available datasets from LLM benchmarks in Section 4.3.

Model	XCOPA				Belebele			
	0-shot	1-shot	2-shot	3-shot	0-shot	1-shot	2-shot	3-shot
Llama-7b	0.53	0.51	0.48	0.52	0.23	0.23	0.23	0.24
LlamaTurk-7b-i	**0.58**	0.51	0.50	0.55	0.24	0.27	0.25	**0.28**
LlamaTurk-7b-c-i	0.52	0.52	0.53	0.50	0.24	0.25	0.23	0.27
LlamaTurk-7b-v-i	0.55	0.53	0.54	0.54	0.24	0.27	0.23	**0.28**

Table 5: **Accuracy scores on benchmark datasets**. The highest scores for each dataset are given in bold.

Task-specific tuning improves the performance of downstream task. We find that task-specific tuning cannot help improve perplexity scores previously. However, our extrinsic evaluation shows that task-specific tuning improves the performance of sentiment analysis. Specifically, we observe that task-specific tuned model (`LllamaTurk-7b-t`) is good at zero-shot inference, suggesting that task-specific instructions provide sufficient knowledge for zero-shot evaluation.

Instruction tuning boosts the performance of downstream task when used together with task-specific tuning. When instruction tuning is employed alone, it has no significant impact on the performance of downstream task. However, we find that the highest accuracy score is obtained when instruction tuning and task-specific tuning are together employed (`LllamaTurk-7b-i-t`). Moreover, `LllamaTurk-7b-i-t` has a better few-shot performance compared to other methods including task-specific tuning.

Continual training can help task-tuning. When continual training is employed alone (`LllamaTurk-7b-c`), we observe no significant improvement in the performance of downstream task. However, the performance is promising when it is used together with task-specific tuning (`LllamaTurk-7b-c-t`). This suggests further examination of continual training with task-specific tuning in different downstream tasks and datasets.

Vocabulary extension has poor downstream performance. Similar to the perplexity experiments, we observe that vocabulary extension has no improvement on the performance of downstream task.

4.3 Benchmark Evaluation

In this subsection, we report the performance results on benchmark datasets. Since LLM evaluation benchmarks mostly include English datasets, we examine multilingual datasets in available LLM benchmarks. For this purpose, we use the Turkish subsets of XCOPA (Ponti et al., 2020) and Belebele (Bandarkar et al., 2023) datasets provided by LM Evaluation Harness (Gao et al., 2023). XCOPA is a benchmark to evaluate the ability of machine learning models to transfer commonsense reasoning. Belebele is a multiple-choice machine reading comprehension dataset, and each question has four multiple-choice. We modify the default prompts given in LM Evaluation Harness to align with our instruction prompting. We provide the inference prompts in Appendix A.4 and A.5.

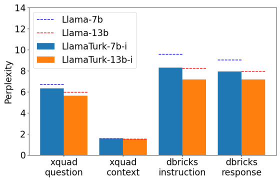
Since the dataset sizes are small, we are not able to apply task-specific tuning in these benchmark datasets. Specifically, we observe almost no change in performance scores when XCOPA's 600 and Belebele's 900 instances are fine-tuned for the Turkish language, while the performance is improved in Section 4.2 with 5k instances. We thereby report the results for instruction tuning and related methods. Table 5 reports the accuracy scores on the XCOPA and Belebele datasets.

The results show that instruction tuning (`LlamaTurk-7b-i`) improves the performance of downstream task in both datasets. However, continual training and vocabulary extension have no significant benefits on the results. The results thereby align with the results of sentiment analysis reported in Section 4.2.

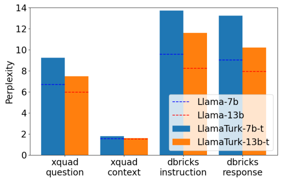
4.4 Model Size

We provide an analysis of the impact of model size on adapting generative LLMs. For this purpose, we employ Llama models with 7b and 13b parameters. Figure 2 shows a histogram depicting the comparison between the fine-tuned models for instruction tuning (`LlamaTurk-7b-i` and `LlamaTurk-13b-i`) and task-specific tuning (`LlamaTurk-7b-t` and `LlamaTurk-13b-t`).

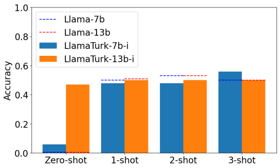
Perplexity is improved by adapting a larger model. In both cases of applying instruction or task-specific tuning, we find that `LlamaTurk-13b` improves perplexity scores in all cases. However, task-specific tuning (`LlamaTurk-13b-t`) is still outperformed by the original Llama model `Llama-13b` in most cases.



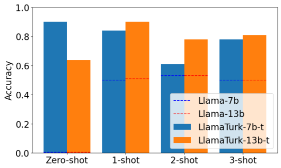
(a) Perplexity of instruction tuning (lower is better)



(b) Perplexity of task-specific tuning (lower is better)

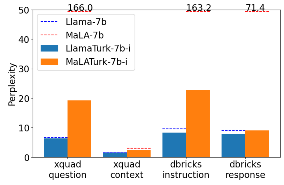


(c) Accuracy of instruction tuning (higher is better)

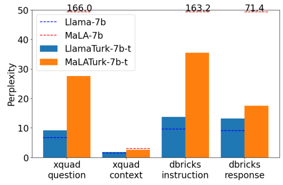


(d) Accuracy of task-specific tuning (higher is better)

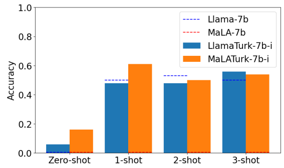
Figure 2: **Model size comparison for adaptation.**



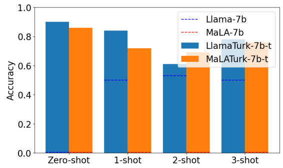
(a) Perplexity of instruction tuning (lower is better).



(b) Perplexity of task-specific tuning (lower is better).



(c) Accuracy of instruction tuning (higher is better).



(d) Accuracy of task-specific tuning (higher is better).

Figure 3: **Multilingual comparison for adaptation.**

Task performance is improved by adapting a larger model when few-shot tuning is applied. We find that `LlamaTurk-13b` improves the performance of downstream task when it is applied with task-specific tuning and few-shot evaluation. On the other hand, the adaptation of a larger model with instruction tuning has no significant impact on the performance of downstream task.

4.5 Multilingual Models

We also provide an analysis for the impact of multilingual generative LLMs on adapting generative LLMs. For this purpose, we fine-tune a multilingual model, MaLA-500 (Lin et al., 2024b). MaLA is developed to cover 534 languages by using vocabulary extension and continual training on Llama2 (Touvron et al., 2023b). Analyzing a multilingual LLM with an enriched vocabulary can provide more insights into LLM adaptation for low-resource languages.

Figure 3 shows a histogram depicting the comparison between the fine-tuned models for instruction tuning (`LlamaTurk-7b-i` and `MaLATurk-7b-i`) and task-specific tuning (`LlamaTurk-7b-t` and `MaLATurk-7b-t`).

Adapting multilingual LLM has no significant improvements. Perplexity and accuracy scores of the original `MaLA-7b` model are improved by adapting `MaLATurk-7b` in both instruction and task-specific tuning. However, the perplexity of adapting a monolingual model `LlamaTurk-7b` is still better than adapting a multilingual model in all cases. Similarly, monolingual adaptation has better accuracy scores of task-specific tuning in most cases. The only benefit of adapting multilingual LLM is observed when instruction tuning is applied.

5 Conclusion

This study examines different methods for adapting English-dominant generative large language models to low-resource languages.

The results show that continual training with raw text can improve perplexity, while vocabulary extension has no significant impact on adaptation performance. We also find that the adaptation with general-purpose instruction tuning has promising results in both perplexity and accuracy scores, while downstream task performance can be boosted by task-specific tuning. Furthermore, adapting a larger model with 13b parameters improves task performance with few-shot tuning. However, we observe no significant improvements by adapting a multilingual model.

In future work, we plan to adapt other open-source language models such as Llama2 (Touvron et al., 2023b) and Gemini (Team et al., 2024) to generalize our results to different models. Other adaptation methods can also be studied such as modification of model architecture since different model layers and tokenization algorithms might change the outcomes.

6 Limitations

This study employs a particular family of generative large language models (Llama and MaLA) for adapting open-source generative monolingual and multilingual LLMs to a low-resource language. Using other generative models might have different results in the experiments. Similarly, we use the Turkish language for the target of adaptation. Other languages might have different experimental results depending on the tuning and inference datasets with prompt examples. We therefore acknowledge the effect of the instruction set and prompting templates in the results.

Moreover, benchmark evaluation is limited to multilingual datasets in this study due to the availability of benchmark datasets for the target language. Lastly, we would like to emphasize the limited hardware resources the experiments were conducted, which restricts using a variety of models including larger sizes (higher than 13b) and different model types (rather than Llama).

7 Ethical Concerns

This study employs a low-resource language, Turkish, and our findings can guide to other researchers studying low-resource languages. We also provide both intrinsic and extrinsic performance evaluations that can be considered for deploying generative LLMs in similar tasks.

To provide transparency, we explain all details regarding text collections used in pretraining and fine-tuning our generative language models. Moreover, we report the details of the models and configurations with hyperparameters.

Since the training corpus of generative LLMs involves a huge amount of raw text from different resources including the world wide web, it is inevitable to observe a risk of cultural and ethical bias towards different individuals and communities in the generated text of the published models in this study (Kasneci et al., 2023; Cetinkaya et al., 2024). Moreover, training texts are contaminated with more problematic biases and polluted with a large amount of synthetic text generated by LLMs (Denning and Rousse, 2024). Possible bias can be removed by filtering the corpus, however, we leave the study of such filtering to future work since it would require a dedicated effort but the scope of this study is to compare the adaptation methods of generative LLMs for low-resource languages.

Lastly, we estimate the carbon footprint of our study based on the energy usage of GPUs. We consider execution time in hours and electrical energy consumption in kWh, and assume that power consumption during training is equal to the maximum power drain of GPUs by operating at maximum power utilization (0.25 MW for 2080Ti, and 0.14 MW for A4000). We assume that 1 MWh is equivalent to 0.439 ton CO2eq[14]. Our estimation ignores the carbon footprint of CPU utilization and the manufacturing costs of the hardware.

Social carbon cost is approximately 50.64, 3.84, and 0.55 kg CO2eq for a single run of continual training, instruction tuning, and task-specific tuning, respectively.

References

Emre Can Acikgoz, Mete Erdogan, and Deniz Yuret. 2024. Bridging the bosphorus: Advancing turkish large language models through strategies for low-resource language adaptation and benchmarking.

01. AI, :, Alex Young, Bei Chen, Chao Li, Chengen Huang, Ge Zhang, Guanwei Zhang, Heng Li, Jiangcheng Zhu, Jianqun Chen, Jing Chang, Kaidong Yu, Peng Liu, Qiang Liu, Shawn Yue, Senbin Yang, Shiming Yang, Tao Yu, Wen Xie, Wenhao Huang, Xiaohui Hu, Xiaoyi Ren, Xinyao Niu, Pengcheng Nie, Yuchi Xu, Yudong Liu, Yue Wang, Yuxuan Cai, Zhenyu Gu, Zhiyuan Liu, and Zonghong Dai. 2024. Yi: Open foundation models by 01.ai.

Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Mérouane Debbah, Étienne Goffinet, Daniel Hesslow, Julien Launay, Quentin Malartic, Daniele Mazzotta, Badreddine Noune, Baptiste Pannier, and Guilherme Penedo. 2023. The falcon series of open language models.

Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. 2019. On the cross-lingual transferability of monolingual representations. *CoRR*, abs/1910.11856.

Murat Aydoğan and Veysel Kocaman. 2023. Trsav1: a new benchmark dataset for classifying user reviews on Turkish e-commerce websites. *Journal of Information Science*, 49(6):1711–1725.

Israel Abebe Azime, Mitiku Yohannes Fuge, Atnafu Lambebo Tonja, Tadesse Destaw Belay, Aman Kassahun Wassie, Eyasu Shiferaw Jada, Yonas Chanie, Walelign Tewabe Sewunetie, and Seid Muhie Yimam. 2024. Enhancing amharic-llama: Integrating task specific and generative datasets. *arXiv preprint arXiv:2402.08015*.

Stephen H. Bach, Victor Sanh, Zheng-Xin Yong, Albert Webson, Colin Raffel, Nihal V. Nayak, Abheesht Sharma, Taewoon Kim, M Saiful Bari, Thibault Fevry, Zaid Alyafeai, Manan Dey, Andrea Santilli, Zhiqing Sun, Srulik Ben-David, Canwen Xu, Gunjan Chhablani, Han Wang, Jason Alan Fries, Maged S. Al-shaibani, Shanya Sharma, Urmish Thakker, Khalid Almubarak, Xiangru Tang, Dragomir Radev, Mike Tian-Jian Jiang, and Alexander M. Rush. 2022. Promptsource: An integrated

[14]https://enerji.gov.tr/evced-cevre-ve-iklim-elektrik-uretim-tuketim-emisyon-faktorleri

development environment and repository for natural language prompts.

Lucas Bandarkar, Davis Liang, Benjamin Muller, Mikel Artetxe, Satya Narayan Shukla, Donald Husa, Naman Goyal, Abhinandan Krishnan, Luke Zettlemoyer, and Madian Khabsa. 2023. The belebele benchmark: a parallel reading comprehension dataset in 122 language variants.

Paweł Budzianowski and Ivan Vulić. 2019. Hello, it's gpt-2–how can i help you? towards the use of pretrained language models for task-oriented dialogue systems. *arXiv preprint arXiv:1907.05774*.

Yusuf Mucahit Cetinkaya, Emre Kulah, Ismail Hakki Toroslu, and Hasan Davulcu. 2024. Targeted marketing on social media: utilizing text analysis to create personalized landing pages. *Soc. Netw. Anal. Min.*, 14(77).

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Mike Conover, Matt Hayes, Ankit Mathur, Jianwei Xie, Jun Wan, Sam Shah, Ali Ghodsi, Patrick Wendell, Matei Zaharia, and Reynold Xin. 2023. Free dolly: Introducing the world's first truly open instruction-tuned llm.

Nicholas Kluge Corrêa, Sophia Falk, Shiza Fatimah, Aniket Sen, and Nythamar de Oliveira. 2024. Teenytinyllama: open-source tiny language models trained in brazilian portuguese.

Yiming Cui, Ziqing Yang, and Xin Yao. 2023. Efficient and effective text encoding for chinese llama and alpaca. *arXiv preprint arXiv:2304.08177*.

Yiming Cui, Ziqing Yang, and Xin Yao. 2024. Efficient and effective text encoding for chinese llama and alpaca.

Peter Denning and B Scot Rousse. 2024. Can machines be in language? *Communications of the ACM*, 67(3):32–35.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Robert M French. 1999. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2023. A framework for few-shot language model evaluation.

Gabriel Lino Garcia, Pedro Henrique Paiola, Luis Henrique Morelli, Giovani Candido, Arnaldo Cândido Júnior, Danilo Samuel Jodas, Luis Afonso, Ivan Rizzo Guilherme, Bruno Elias Penteado, and João Paulo Papa. 2024. Introducing bode: A fine-tuned large language model for portuguese prompt-based task. *arXiv preprint arXiv:2401.02909*.

Kshitij Gupta, Benjamin Thérien, Adam Ibrahim, Mats L Richter, Quentin Anthony, Eugene Belilovsky, Irina Rish, and Timothée Lesort. 2023. Continual pre-training of large language models: How to (re) warm your model? *arXiv preprint arXiv:2308.04014*.

Oskar Holmström and Ehsan Doostmohammadi. 2023. Making instruction finetuning accessible to non-English languages: A case study on Swedish models. In *Proceedings of the 24th Nordic Conference on Computational Linguistics (NoDaLiDa)*, pages 634–642, Tórshavn, Faroe Islands. University of Tartu Library.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Jaavid Aktar Husain, Raj Dabre, Aswanth Kumar, Jay Gala, Thanmay Jayakumar, Ratish Puduppully, and Anoop Kunchukuttan. 2024. Romansetu: Efficiently unlocking multilingual capabilities of large language models models via romanization.

Pau Rodriguez Inserte, Mariam Nakhlé, Raheel Qader, Gaetan Caillaut, and Jingshu Liu. 2024. Large language model adaptation for financial sentiment analysis.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b.

Daniel Jurafsky and James H. Martin. 2009. *Speech and Language Processing (2nd Edition)*. Prentice-Hall, Inc., USA.

Enkelejda Kasneci, Kathrin Seßler, Stefan Küchemann, Maria Bannert, Daryna Dementieva, Frank Fischer, Urs Gasser, Georg Groh, Stephan Günnemann, Eyke Hüllermeier, et al. 2023. Chatgpt for good? on opportunities and challenges of large language models for education. *Learning and individual differences*, 103:102274.

H. Toprak Kesgin, M. Kaan Yuce, Eren Dogan, M. Egemen Uzun, Atahan Uz, H. Emre Seyrek, Ahmed Zeer, and M. Fatih Amasyali. 2024. Introducing cosmosgpt: Monolingual training for turkish language models.

Guneet Singh Kohli, Shantipriya Parida, Sambit Sekhar, Samirit Saha, Nipun B Nair, Parul Agarwal, Sonal Khosla, Kusumlata Patiyal, and Debasish Dhal. 2023. Building a llama2-finetuned llm for odia language utilizing domain knowledge instruction set.

Hele-Andra Kuulmets, Taido Purason, Agnes Luhtaru, and Mark Fishel. 2024. Teaching llama a new language through cross-lingual knowledge transfer.

Celio Larcher, Marcos Piau, Paulo Finardi, Pedro Gengo, Piero Esposito, and Vinicius Caridá. 2023. Cabrita: closing the gap for foreign languages.

Chen-An Li and Hung-Yi Lee. 2024. Examining forgetting in continual pre-training of aligned large language models. *arXiv preprint arXiv:2401.03129*.

Peiqin Lin, Shaoxiong Ji, Jörg Tiedemann, André FT Martins, and Hinrich Schütze. 2024a. Mala-500: Massive language adaptation of large language models. *arXiv preprint arXiv:2401.13303*.

Peiqin Lin, Shaoxiong Ji, Jörg Tiedemann, André F. T. Martins, and Hinrich Schütze. 2024b. Mala-500: Massive language adaptation of large language models.

Bing Liu. 2012. *Sentiment Analysis and Opinion Mining*. Morgan & Claypool Publishers.

El Moatez Billah Nagoudi, Muhammad Abdul-Mageed, AbdelRahim Elmadany, Alcides Alcoba Inciarte, and Md Tawkat Islam Khondaker. 2023. Jasmine: Arabic gpt models for few-shot learning.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Louis Owen, Vishesh Tripathi, Abhay Kumar, and Biddwan Ahmed. 2024. Komodo: A linguistic expedition into indonesia's regional languages.

Edoardo M. Ponti, Goran Glavaš, Olga Majewska, Qianchu Liu, Ivan Vulić, and Anna Korhonen. 2020. XCOPA: A multilingual dataset for causal commonsense reasoning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Libo Qin, Qiguang Chen, Yuhang Zhou, Zhi Chen, Yinghui Li, Lizi Liao, Min Li, Wanxiang Che, and Philip S. Yu. 2024. Multilingual large language model: A survey of resources, taxonomy and frontiers.

Yujia Qin, Jiajie Zhang, Yankai Lin, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2022. Elle: Efficient lifelong pre-training for emerging data. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2789–2810.

Vipul Raheja, Dhruv Kumar, Ryan Koo, and Dongyeop Kang. 2023. Coedit: Text editing by task-specific instruction tuning.

Andrea Santilli and Emanuele Rodolà. 2023. Camoscio: An italian instruction-tuned llama. *arXiv preprint arXiv:2307.16456*.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, and Matthias Gallé et al. 2022. BLOOM: A 176b-parameter open-access multilingual language model.

Mike Schuster and Kaisuke Nakajima. 2012. Japanese and korean voice search. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5149–5152.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Oleh Shliazhko, Alena Fenogenova, Maria Tikhonova, Vladislav Mikhailov, Anastasia Kozlova, and Tatiana Shavrina. 2023. mgpt: Few-shot learners go multilingual.

Lichao Sun, Yue Huang, Haoran Wang, Siyuan Wu, Qihui Zhang, Chujie Gao, Yixin Huang, Wenhan Lyu, Yixuan Zhang, Xiner Li, et al. 2024. Trustllm: Trustworthiness in large language models. *arXiv preprint arXiv:2401.05561*.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M. Dai, Anja Hauth, Katie Millican, et al. 2024. Gemini: A family of highly capable multimodal models.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023a. Llama: Open and efficient foundation language models.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti

Bhosale, et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Gökçe Uludoğan, Zeynep Yirmibeşoğlu Balal, Furkan Akkurt, Melikşah Türker, Onur Güngör, and Susan Üsküdarlı. 2024. Turna: A Turkish encoder-decoder language model for enhanced understanding and generation.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. *SuperGLUE: a stickier benchmark for general-purpose language understanding systems*. Curran Associates Inc., Red Hook, NY, USA.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Wenxuan Wang, Zhaopeng Tu, Chang Chen, Youliang Yuan, Jen tse Huang, Wenxiang Jiao, and Michael R. Lyu. 2023. All languages matter: On the multilingual safety of large language models.

Yizhong Wang, Hamish Ivison, Pradeep Dasigi, Jack Hessel, Tushar Khot, Khyathi Chandu, David Wadden, Kelsey MacMillan, Noah A Smith, Iz Beltagy, et al. 2024. How far can camels go? exploring the state of instruction tuning on open resources. *Advances in Neural Information Processing Systems*, 36.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Frank F. Xu, Uri Alon, Graham Neubig, and Vincent Josua Hellendoorn. 2022. A systematic evaluation of large language models of code. In *Proceedings of the 6th ACM SIGPLAN International Symposium on Machine Programming*, MAPS 2022, page 1–10, New York, NY, USA. Association for Computing Machinery.

Jingfeng Yang, Hongye Jin, Ruixiang Tang, Xiaotian Han, Qizhang Feng, Haoming Jiang, Shaochen Zhong, Bing Yin, and Xia Hu. 2023a. Harnessing the power of llms in practice: A survey on chatgpt and beyond. *ACM Transactions on Knowledge Discovery from Data*.

Zijian Győző Yang, László János Laki, Tamás Váradi, and Gábor Prószéky. 2023b. Mono-and multilingual gpt-3 models for hungarian. In *International Conference on Text, Speech, and Dialogue*, pages 94–104. Springer.

Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, and Guoyin Wang. 2024. Instruction tuning for large language models: A survey.

Xiang Zhang, Senyu Li, Bradley Hauer, Ning Shi, and Grzegorz Kondrak. 2023a. Don't trust ChatGPT when your question is not in English: A study of multilingual abilities and types of LLMs. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7915–7927, Singapore. Association for Computational Linguistics.

Zheng Zhang, Chen Zheng, Da Tang, Ke Sun, Yukun Ma, Yingtong Bu, Xun Zhou, and Liang Zhao. 2023b. Balancing specialized and general skills in llms: The impact of modern tuning and data strategy.

Jun Zhao, Zhihao Zhang, Luhui Gao, Qi Zhang, Tao Gui, and Xuanjing Huang. 2024. Llama beyond english: An empirical study on language capability transfer.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. 2023a. A survey of large language models.

Xujiang Zhao, Jiaying Lu, Chengyuan Deng, Can Zheng, Junxiang Wang, Tanmoy Chowdhury, Li Yun, Hejie Cui, Zhang Xuchao, Tianjiao Zhao, et al. 2023b. Domain specialization as the key to make large language models disruptive: A comprehensive survey. *arXiv preprint arXiv:2305.18703*.

Jiawei Zheng, Hanghai Hong, Xiaoli Wang, Jingsong Su, Yonggui Liang, and Shikai Wu. 2024. Fine-tuning large language models for domain-specific machine translation.

Ahmet Üstün, Viraat Aryabumi, Zheng-Xin Yong, Wei-Yin Ko, Daniel D'souza, Gbemileke Onilude, Neel Bhandari, Shivalika Singh, Hui-Lee Ooi, Amr Kayid, Freddie Vargus, Phil Blunsom, Shayne Longpre, Niklas Muennighoff, Marzieh Fadaee, Julia Kreutzer, and Sara Hooker. 2024. Aya model: An instruction finetuned open-access multilingual language model.

A Appendix

A.1 Instruction Fine-tuning Prompt

The prompt used in instruction tuning is given as follows (translated prompt is given in parenthesis).

```
Aşağıda, daha geniş bir bağlam sağlayan
girdiyle birlikte bir görevi açıklayan
talimat bulunmaktadır. Talimatı yeterince
sağlayan bir çıktı yaz.
(Below is an instruction explaining a task
with  input that provides more context.
```

```
Write an output satisfying the instruction)

### Talimat (Instruction):
[INSTRUCTION]

### Girdi (Input):
[INPUT]

### Çıktı (Output):
[OUTPUT]
```

A.2 Task-Specific Fine-tuning Prompt

The prompt used in task-specific (sentiment analysis) fine-tuning is given as follows (translated prompt is given in parenthesis).

```
Aşağıda bir görevi açıklayan talimat
bulunmaktadır. Talimatı yeterince
sağlayan bir çıktı yaz.
(Below are instructions describing a task.
Write an output that satisfying
the instruction)

### Talimat:
Lütfen verilen yorumun olumlu ya da
olumsuz olduğunu çıktı olarak belirtin.
(Please indicate whether the given comment
is positive or negative.)

### Yorum (Comment):
[INPUT]

### Çıktı (Output):
[OUTPUT]
```

A.3 Task-Specific Inference Prompt

For sentiment analysis, the prompt used in zero-shot inference is the same as the prompt used for task-specific fine-tuning given in A.2. Few-shot prompting (one-shot for example) is given as follows (translated prompt is given in parenthesis).

```
Aşağıda bir görevi açıklayan talimat
bulunmaktadır. Talimatı yeterince sağlayan
bir çıktı yaz.
(Below are instructions describing a task.
Write an output satisfying the instruction)
### Talimat (Instruction):
Lütfen verilen yorumun olumlu ya da
olumsuz olduğunu çıktı olarak belirtin.
(Please indicate whether the given comment
is positive or negative.)

### Yorum (Comment):
çok güzel, sağlıklı, temiz, ferah
(very beautiful, healthy, clean, spacious)

### Çıktı (Output):
olumlu
(positive)

### Talimat (Instruction):
Lütfen verilen yorumun olumlu ya da
olumsuz olduğunu çıktı olarak belirtin.
(Please indicate whether the given comment
is positive or negative.)

### Yorum (Comment):
[INPUT]

### Çıktı (Output):
[OUTPUT]
```

A.4 XCOPA Inference Prompt

Few-shot prompting (one-shot for example) is given as follows (translated prompt is given in parenthesis).

```
Aşağıda bir görevi açıklayan talimat
bulunmaktadır. Talimatı yeterince
sağlayan bir çıktı yaz.
(Below are instructions describing a task.
Write an output satisfying the instruction)

### Talimat (Instruction):
Verilen cümlenin sebebi nedir?
(What is the reason for the given sentence?)
Kadın kötü bir ruh halindeydi bu yüzden
(The woman was in a bad mood so)

### Girdi (Input):
arkadaşıyla biraz konuştu.
(she talked to her friend for a while.)
arkadaşına onu yalnız bırakmasını söyledi.
(she told her friend to leave her alone.)

### Çıktı (Output):
Kadın kötü bir ruh halindeydi bu yüzden
arkadaşına onu yalnız bırakmasını söyledi.
(The woman was in a bad mood so she told
her friend to leave her alone.)

Aşağıda bir görevi açıklayan talimat
bulunmaktadır. Talimatı yeterince sağlayan
bir çıktı yaz.
```

```
(Below are instructions describing a task.
Write an output satisfying the instruction)

### Talimat (Instruction):
Verilen cümlenin sebebi nedir?
(What is the reason for the given sentence?)
[INPUT]

### Girdi (Input):
[OPTION1]
[OPTION2]

### Çıktı (Output):
Ürün balonlu naylonla paketlenmişti
bu yüzden [OUTPUT]
(The product was packaged with
bubble wrap so [OUTPUT])
```

A.5 Belebele Inference Prompt

Few-shot prompting (one-shot for example) is given as follows (translated prompt is given in parenthesis).

```
Aşağıda bir görevi açıklayan talimat
bulunmaktadır. Talimatı yeterince
sağlayan bir çıktı yaz.
(Below are instructions describing a task.
Write an output satisfying the instruction)

### Talimat (Instruction):
Tüm notalara doğru şekilde basmaya devam
ederken elinizin mümkün olduğu kadar
rahat olduğundan emin olun - aynı zamanda
parmaklarınızla fazladan hareketler
yapmamaya çalışın. Bu şekilde kendinizi
olabildiğince az yormuş olacaksınız.
Unutmayın ki piyanoda olduğu gibi daha
fazla ses için tuşlara çok güçlü
vurmanıza gerek yoktur. Akordeon
üzerinde, ekstra hacim elde etmek için
körüğü daha fazla basınç veya hızda
kullanırsınız. Akordeonu çalarken
aşağıdakilerden hangisi sesin
yükselmesini sağlar?
(Make sure your hand is as relaxed as
possible while still hitting all the
notes correctly - at the same time,
try not to make extra movements with
your fingers. This way, you will tire
yourself as little as possible.
Remember that you don't need to hit
the keys too hard to get more sound,
like on the piano. On the accordion,
you use the bellows with more pressure
or speed to get extra volume.
Which of the following makes the sound
rise when playing the accordion?)

### Girdi (Input):
A: Daha fazla hız (more speed)
B: Daha fazla güç (more power)
C: Daha az basınç (less pressure)
D: Daha az parmak hareketi
(less finger movement)

### Çıktı (Output):
A

Aşağıda bir görevi açıklayan talimat
bulunmaktadır. Talimatı yeterince
sağlayan bir çıktı yaz.
(Below are instructions describing a task.
Write an output satisfying the instruction)

### Talimat (Instruction):
Tüm notalara doğru şekilde basmaya devam
ederken elinizin mümkün olduğu kadar
rahat olduğundan emin olun - aynı zamanda
parmaklarınızla fazladan hareketler
yapmamaya çalışın. ... Akordeonu çalarken
aşağıdakilerden hangisi sesin
yükselmesini sağlar?
(Make sure your hand is as relaxed as
possible while still hitting all the
notes correctly - at the same time,
try not to make extra movements with
your fingers. ... Which of the
following makes the sound rise
when playing the accordion?)

### Girdi (Input):
[OPTION1]
[OPTION2]
[OPTION3]
[OPTION4]

### Çıktı (Output):
[OUTPUT]
```